1-2009

Using Sunflower Plots and Classification Trees to Study Typeface Legibility

Edgar C. Merkle
Wichita State University

Barbara S. Chaparro
Wichita State University, chaparb1@erau.edu

CS-BIGS 2(2): 92-98
http://www.bentley.edu/csbigs/vol2-2/merkle.pdf

Using Sunflower Plots and Classification Trees to Study Typeface Legibility

Edgar C. Merkle and Barbara S. Chaparro
Wichita State University, USA

This article describes the application of sunflower plots and classification trees to the study of onscreen typeface legibility. The two methods are useful for describing high-dimensional data in an intuitive manner, which is crucial for interacting with both the typographers who design the typefaces and the practitioners who must make decisions about which typeface to use for specific applications. Furthermore, classification trees help us make specific recommendations for how much of a character attribute is "enough" to make it legible. We present examples of sunflower plots and classification trees using data from a recent typeface legibility experiment, and we present R code for replicating our analyses. Some familiarity with classification trees and logistic regression will be helpful to the reader.

1. Using sunflower plots and classification trees to study typeface legibility

What makes one typeface more legible on a computer screen than another? Can we determine which attributes of a specific character are most important for onscreen legibility? These questions have important implications for the design of legible typefaces. Legibility is especially critical in situations where single characters must be discerned quickly. For example, air traffic controllers must be able to identify aircraft information such as aircraft type, affiliation, speed, and altitude from a short multi-character code. Everyday computer users must also be able to easily identify single characters when reviewing spreadsheet data or entering account usernames and passwords.

The above issues were of interest to the Advanced Reading Technology team at Microsoft, which develops and researches new typefaces. With support from this team, we conducted a series of experiments to examine the legibility of various typefaces. Given that the results of these studies needed to be understood by typographers, practitioners, and onscreen designers, we wanted to make the statistical results of our experiment as intuitive as possible.

In this article, we discuss two statistical procedures that proved very useful for describing our experimental results: sunflower plots and classification trees. We used the former procedure to display legibility results for specific typefaces, while we used the latter procedure to make recommendations about how to design specific characters. We begin the paper by describing the experimental design, and we then outline our use of sunflower plots and classification trees. Finally, we include R code and data for replicating our analyses.

2. Experimental Methods

The data that we describe come from a single character legibility experiment (Chaparro, Merkle, & Fox, under review). Experimental participants were required to identify individual characters that were flashed briefly on a computer screen (34 msec). This does not mimic a

reading scenario, where the context of characters (e.g., surrounding characters) plays an important role, but it is still informative of relative legibility across typefaces and across characters. Ten participants were tested on 47 characters across 20 typefaces; each character×typeface combination was presented three times. The presentation order of typefaces and characters within each typeface was randomized. The test characters, all of which were presented in 10-point font size, were twenty-six lowercase letters, numerals 0–9, and eleven common symbols (÷ = + ? % ± $ # @ & !). This was generally a long experiment for the participants, and testing was broken up over three days.

In addition to recording participants' responses on each trial, researchers measured many attributes of each presented character. These attributes included measures such as the height of a character, the width of a character, and whether or not the character has a serif (serifs are small strokes at the ends of the lines that make up a character). The goals of the study were: (1) to identify the attributes that are most related to the correct identification of individual characters; and (2) to determine the legibility of different typefaces.

While we initially considered some interesting, complex statistical models for the data (e.g., hierarchical logistic regression models), the results of these models could be difficult to describe to an audience with less statistical knowledge. Furthermore, these models yield little information about recommendations for each attribute. For example, logistic regression may tell us that the height of an "s" is related to the correct identification of an "s," but it would not tell us what height is "tall enough." There were also some technical problems with the use of regression. For example, there were many predictor variables (up to 11, depending on the character), and these predictor variables were often related to one another. Thus, it was difficult to find subsets of attributes that were most important for identifying a specific character. The statistical tools that we describe below proved to be more compatible with our audience and with the goals of the study.

3. Sunflower Plots

3.1 Introduction

Sunflower plots were originally introduced by Chambers et al. (1983) and Cleveland and McGill (1984) as an alternative to regular scatter plots (while the Chambers et al. book has an earlier publication date, the authors of that book cite Cleveland and McGill as the creators of the sunflower plot).

In sunflower plots, each continuous variable (say, x and y) is grouped into small bins (say, $x(1),x(2),\ldots,x(m)$ and $y(1),y(2),\ldots y(n)$). The number of observations that fall in each $x(i) \times y(j)$-bin combination are calculated, and sunflowers are created for each combination containing greater than one observation. For every observation within a bin, a petal is added to the corresponding sunflower (if there is only one observation, a point is displayed instead of a sunflower).

Sunflower plots are intended to alleviate the problem of overlap among points in regular scatter plots. Overlap between points often occurs for large datasets or for small datasets with low variability, making it difficult to determine the relative frequencies of observations in different parts of the plot (there could be many observations at a single point, but only one point is actually displayed). In contrast, there is no overlap in sunflower plots: if many observations are clustered together, the sunflower for the corresponding bin simply has more petals than other sunflowers. This yields a display that allows the observer to quickly discern the location of the majority of observations within a dataset. For the typeface legibility study, we extended the sunflower plots to display categorical variables.

3.2 Application to Typeface Data

Sunflower plots were used to intuitively display the results for each individual typeface. Two such plots are displayed in Figures 1 and 2. Figure 1 displays results for the Verdana typeface, while Figure 2 displays results for the Garamond typeface. On each plot, the x-axis contains the presented characters and the y-axis contains the characters reported by the participants (i.e., the "predicted" characters). For each font, there were 10 subjects×47 characters×3 trials per character=1410 observations. Participants correctly identified Verdana characters 97% of the time, while they correctly identified Garamond characters 93% of the time. The diagonals have been removed from the plots so that only error responses are displayed. People are generally good at identifying characters, so the proportion of correct responses is always relatively high. Thus, the diagonal is very distracting because it contains sunflowers with many petals.

The grids within each plot signify different types of confusions: "SS" represents trials where a symbol was confused with a different symbol. "NS" represents trials

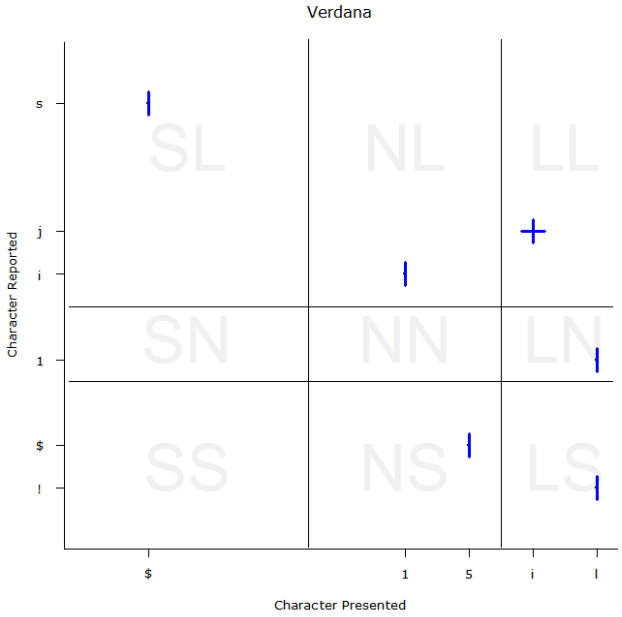

Figure 1. Sunflower plot for the identification of Verdana characters. Characters in the plot labels are displayed in the Verdana typeface

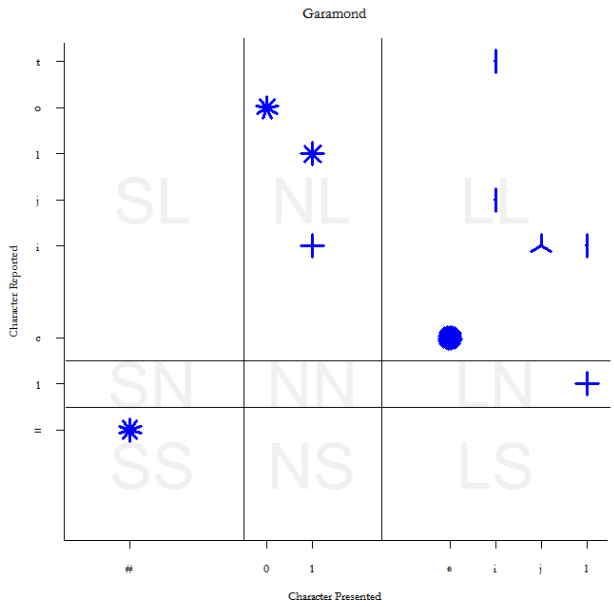

Figure 2. Sunflower plot for the identification of Garamond characters. Characters in the plot labels are displayed in the Garamond typeface

where a number was confused with a symbol. "LS" represents trials where a letter was confused with a symbol. Other boxes in the grid are defined similarly, with "S" in place of Symbol, "N" in place of Number and "L" in place of Letter. In each plot, the axis labels are displayed in the actual typeface that was tested. This allows the reader to view the character in its presented form and helps to explain why a character may be problematic.

Verdana was found to be one of the most legible typefaces of the twenty tested. However, there were a small number of confusions. In examining Figure 1, the confusions are quite clear; namely, the letter 'i' was confused most often with the letter 'j'. Garamond was found to be one of the least legible typefaces tested. Again, the problematic characters are readily apparent in the sunflower plot (Figure 2). The number '0' was confused with the letter 'o', the '#' sign was confused with the '=' sign, the number '1' was confused with the letter 'l', and the letter 'e' was confused with the letter 'c'.

3.3 Discussion

In comparing the two plots, it is apparent that performance with Verdana was much better than performance with Garamond. Not only are there more sunflowers present on the Garamond plot (indicating more characters confused) but also the density of the sunflowers is much greater, indicating a greater number of total confusions.

The sunflower plots of Example 2 are an excellent method for displaying character misclassifications made by human observers. While the data could have been displayed in tabular format, the plots are better at directing reader attention to the least legible characters and their respective degree of confusion (e.g., Gelman, Pasarica, & Dodhia, 2002). Appendix A contains details on how we created these plots in R. These details are not trivial, and we hope that other researchers can benefit from them.

4. Classification Trees

4.1 Introduction

Classification and regression trees, also known as binary recursive partitioning, are computationally-intensive, nonparametric statistical methods that can be used in regression-like situations. "Classification" refers to a tree-building procedure with a categorical response variable, whereas "regression" refers to a tree-building procedure with a continuous response variable. We focus on classification trees here.

There are many differences between regression models and classification trees. Classification trees output a

decision tree that is used to predict a response variable from the predictor variables. Regression models, on the other hand, output a linear equation that minimizes the discrepancy between the model and the data. While both methods can be used for prediction, classification tree output tends to be more intuitive (e.g., Breiman, 2001).

The computational algorithms underlying classification trees are too complex to thoroughly describe in this paper. Generally, classification trees attempt to predict a response variable by sequentially splitting the data into two groups. The splits are based on values of the predictor variables, and they are chosen to maximize predictive accuracy of the response variable. A major issue of this procedure involves the decision of when to stop splitting up the data. For example, assume that we split the data into two groups. We could decide that those two groups are enough, or we could split those two groups into more subgroups. The general strategy underlying many modern classification tree methods is to split the data into many subgroups (too many to be useful). After all the subgroups are obtained, there is a "pruning" step in which less-important subgroups are deleted. The pruning process is designed to yield the smallest tree that can accurately predict the response variable.

For more details, the interested reader is referred to full-text treatments by Breiman et al. (1984) and Zhang and Singer (1999), as well as shorter chapters/articles by Clark and Pregibon (1992), Merkle and Shaffer (under review), and Ripley (1996).

4.2 Application to Typeface Data

As applied to the typeface legibility data, we built a separate decision tree for each character. There were 10 participants×20 typefaces×3 trials=600 observations per character. The goal of the analysis was to determine attributes of the character (height, width, etc.) that influence the character's legibility. For each character within each of the twenty typefaces, approximately 10 attributes were measured (different attributes are relevant to different characters). For a specific character, the attributes across all twenty typefaces were then used as predictor variables in a classification tree analysis. The response variable in this analysis was trial-by-trial accuracy (a dichotomous variable). The outcome of the analysis was a decision tree telling us which attributes were associated with greater/lesser legibility.

In building the classification trees, we ran into the issue of high proportions correct for some characters. That is, experimental participants were able to achieve near 100% accuracy in identifying some characters regardless of the typeface. When this happened, we were unable to build classification trees; more generally, it is impossible to examine the impact of character attributes on legibility when all participants are near the ceiling in accuracy. While it is possible to use degraded characters in these situations (i.e., decreased point size or increased blurriness) to avoid ceiling effects, the typeface designers strongly preferred that we use non-degraded characters (i.e., displaying the characters "in the way that they were designed to be seen").

Figure 3. Classification tree for the letter e
Notes: Starting at the top of the tree, different branches are followed depending on specific character attributes. Once an endpoint is reached, a prediction is made (0=incorrect, 1=correct). Numbers separated by '/' indicate the observed number of incorrect and correct identifications, respectively, within a node.

Classification trees for the letter 'e' and number '0' appear in Figures 3 and 4. R code for building and plotting these trees appears in Appendix B. Starting with ten attributes of the letter e and nine attributes of the number 0, the trees have selected a small number of attributes that influence legibility. Focusing on the letter 'e', midline was selected as the only attribute influencing legibility. Midline is the height of the horizontal line in the letter 'e', relative to the overall height. Figure 3 shows that small (below .61) values of midline were associated with high legibility (92% correct identification), whereas larger values of midline were associated with low legibility (10% correct identification). Large values of midline mean that the e's horizontal line is relatively low in the character, making it confusable with the letter 'o' or number '0'. While the value of .61 estimates the threshold value at which midline impacts legibility, there is nothing special about that specific number. It is just the average of two observed midline values, one larger than .61 and one smaller than .61. We could have alternatively chosen .60 (say) as the threshold value, with no change in the results.

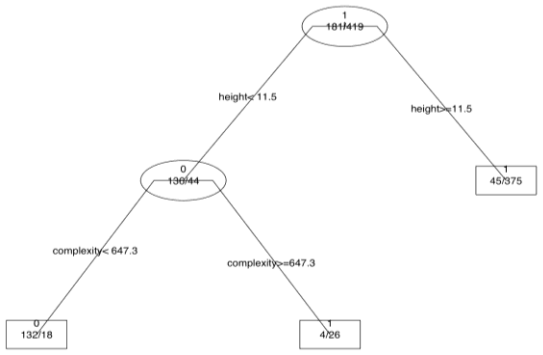

Figure 4. Classification tree for the number 0
Notes. Starting at the top of the tree, different branches are followed depending on specific character attributes. Once an endpoint is reached, a prediction is made (0=incorrect, 1=correct). Numbers separated by '/' indicate the observed number of incorrect and correct identifications, respectively, within a node.

Focusing on the number '0' (Figure 4), the measures of height and complexity both had an effect on legibility. Short zeroes were confused more often than tall zeroes (24% versus 89% correct identification), likely due to the fact that short zeroes look like the letter 'o'. For the shorter zeroes, "complexity" also played a role in legibility. Complexity is defined as perimeter^2/ink area (e.g., Pelli, Burns, Farell, & Moore-Page, 2006). Thus, Figure 4 shows that shorter zeroes were still legible if their ratio of perimeter to ink area was large.

The above two trees were some of the most compelling results that we obtained. The tree for the letter 'l', displayed in Figure 5, is an example of a less-compelling result. This tree is more complex due to the multiple branches, and the proportions correct in the end branches are not as disparate as those for the previous trees. We might conclude that short, wide l's can be problematic, depending on their weight (the weight is the darkness (blackness) of a character, independent of its size). If these l's have weights between 7.4 and 7.8, or above 10.2, then legibility is poor (44% correct for l's of this type, vs. 73% correct for other l's). However, the pruning results (not shown) imply that these branches are less useful for predicting legibility than those for '0' and 'e'.

4.3 Discussion

For individual characters, classification trees were able to quickly identify important character attributes related to legibility and yield information about the nature of the attributes' relationships to legibility. Expanding on the second point, the classification trees made specific recommendations on how much of an attribute is "enough" to improve legibility. This information can be very useful to the typographers designing characters. In contrast, a logistic regression model might tell us that height is related to legibility, but it would not immediately tell us anything about which heights lead to increased or decreased legibility. To resolve this issue, it would be possible to employ a logistic regression model with threshold values for various attributes (e.g., a dummy variable that equals 0 if midline is below .6 and 1 if midline is above .6). However, these threshold values would likely have to be set by hand, and variable selection would have to occur prior to the setting of the thresholds.

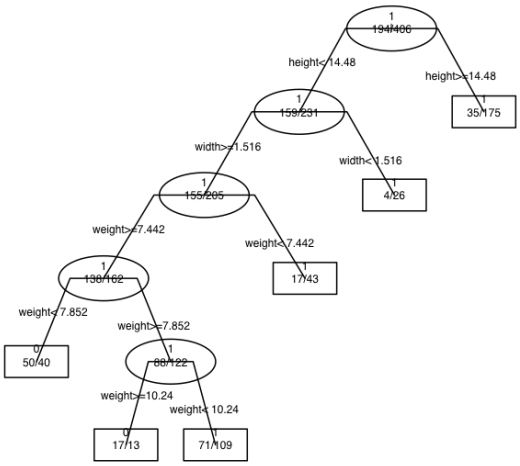

Figure 5. Classification tree for the letter l
Notes. Starting at the top of the tree, different branches are followed depending on specific character attributes. Once an endpoint is reached, a prediction is made (0=incorrect, 1=correct). Numbers separated by '/' indicate the observed number of incorrect and correct identifications, respectively, within a node.

5. Conclusions

In this paper, we have illustrated the application of sunflower plots and classification trees to the study of typeface legibility. Sunflower plots were used to examine the legibility of various typefaces. Classification trees, on the other hand, were used to examine attributes affecting the legibility of a specific character. Taken together, our choice of analyses reflects the fact that we were communicating our results to an audience who may not have a background in statistics or research training. Thus, our results had to be as intuitive and concrete as possible.

We found sunflower plots and classification trees to fulfill these goals, and we recommend that researchers in similar situations explore the use of these methods. Furthermore, as shown in the appendices, the analyses are straightforward to implement in R (though tailoring the graphs can be time consuming).

Appendix: Sunflower Plot Details

In this appendix, we provide details on creating sunflower plots. We begin with some general notes on the plots, and we then provide code and data for generating the Verdana plot.

For an excellent, detailed background on R plots, see Murrell (2006).

General Notes.

- We created the plots in R 2.7.0 Pre-release on Windows XP. We worked in a Windows environment so that we could display the axis labels in Microsoft typefaces.

- Because we used Windows to create the plots, the R commands work best in a Windows environment. Modifications for Linux environments are relatively straightforward and can be obtained by emailing the first author.

- Correct responses have been eliminated from the plots. Had we kept correct responses, we would have had a diagonal of thick sunflowers. We judged this to be too distracting. We also removed characters that were confused three or fewer times, so that we could highlight the characters that were confused most often.

- To insert specific Microsoft typefaces in the plot labels, we had to edit the Rdevga file to include the typeface names. Starting in the R directory, this file can be found in the etc/ subdirectory. Editing of the file simply involves adding a line that names the desired typeface (see the file itself for the specific line requirements).

Files.

The included files for creating sunflower plots are:

- sunflower.R: Main code file that creates the plot and makes use of the other files.

- verdana.dat: Main data file that contains three columns, named x, y, and number. x and y contain numbers that stand for different characters (the character names are in the names files). The i^{th} row of verdana.dat contains the number of times that character x was presented and character y was reported. To create this data format from trial-by-trial data files, see the xyTable() command.

- xnames.txt, ynames.txt: Contains information on the number codes in verdana.dat.

R Code.

Assuming that the above files are contained in the working directory, the following commands can be used to create the plot. The plot will not be displayed in Verdana unless Verdana is the eighth typeface in the Rdevga file (see notes above).

```
dat <- read.delim("verdana.dat")
xnames <- scan("xnames.txt",what="factor",
               blank.lines.skip=FALSE)
ynames <- scan("ynames.txt",what="factor",
               blank.lines.skip=FALSE)

source("sunflower.R")

sunflower(dat,xnames,ynames,cutoff=2)
```

Creating Classification Trees

```
# Must install the rpart package first.
# Enter the following command and follow prompts

# (only need to do this once):
install.packages("rpart")

# Load package:
library(rpart)

# Read data into R:
zero.dat <- read.delim("0-data.txt")
e.dat <- read.delim("e-data.txt")

# Clarify that "corr" columns are dichotomous
# (as opposed to continuous):
zero.dat$corr <- as.factor(zero.dat$corr)
e.dat$corr <- as.factor(e.dat$corr)

# Build trees:
zero.tr <- rpart(corr ~ ., data=zero.dat[,2:11])

e.tr <- rpart(corr ~ ., data=e.dat[,2:11])

# Plot trees:
plot(zero.tr); text(zero.tr)
plot(e.tr); text(e.tr)
```

The previous commands can be used to read in the typeface data and build classification trees for the number 0 and the letter e.

REFERENCES

Breiman, L. (2001). Statistical modeling: The two cultures. *Statistical Science*, *16*, 199-231.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth.

Chambers, J. M., Cleveland, W. S., Kleiner, B., & Tukey, P. A. (1983). *Graphical methods for data analysis*. Boston: Duxbury.

Chaparro, B. S., Merkle, E. C., & Fox, D. E. (under review). Examination of the features that influence the legibility of onscreen typefaces.

Clark, L. A., & Pregibon, D. (1992). Tree-based models. In J. M. Chambers & T. J. Hastie (Eds.), *Statistical models in S* (p. 377-419). Pacific Grove, CA: Wadsworth.

Cleveland, W. S., & McGill, R. (1984). The many faces of a scatterplot. *Journal of the American Statistical Association*, *79*, 807-822.

Gelman, A., Pasarica, C., & Dodhia, R. (2002). Let's practice what we preach: Turning tables into graphs. *The American Statistician*, *56*, 121-130.

Merkle, E. C., & Shaffer, V. A. (under review). Binary recursive partitioning methods with application to psychology.

Murrell, P. (2006). *R Graphics*. Boca Raton, FL: Chapman & Hall.

Pelli, D. G., Burns, C. W., Farell, B., & Moore-Page, D. C. (2006). Feature detection and letter identification. *Vision Research*, *46*, 4646–4674.

Ripley, B. D. (1996). *Pattern recognition and neural networks*. New York: Cambridge.

Zhang, H., & Singer, B. (1999). *Recursive partitioning in the health sciences*. New York: Springer.

Author Note

The study was funded by a grant from the Advanced Reading Technology team at Microsoft Corporation. Communication regarding this paper can be addressed to Edgar Merkle, Department of Psychology, 1845 Fairmount, Wichita, KS 67260-0034

Correspondence: edgar.merkle@wichita.edu